git - the simple guide

just a simple guide for getting started with git. no deep shit ;)

Tweet 4,747

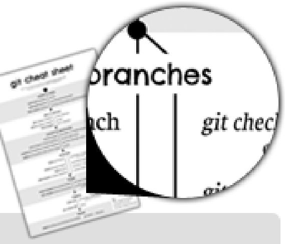
by Roger Dudler

credits to @tfnico, @fhd and Namics

deutsch, español, français, indonesian, italiano, nederlands, polski, português, русск

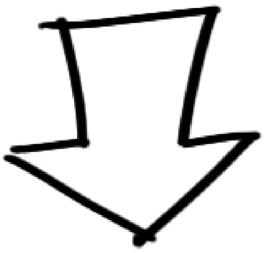
မြန်မာ, 日本語, 中文, 한국어 Vietnamese

please report issues on github



download the cheat sheet now. it's free!

want a simple but powerful git client for your mac?



setup

Download git for OSX

Download git for Windows

Download git for Linux

create a new repository

create a new directory, open it and perform a

```
git init
```

to create a new git repository.

checkout a repository

create a working copy of a local repository by running the command

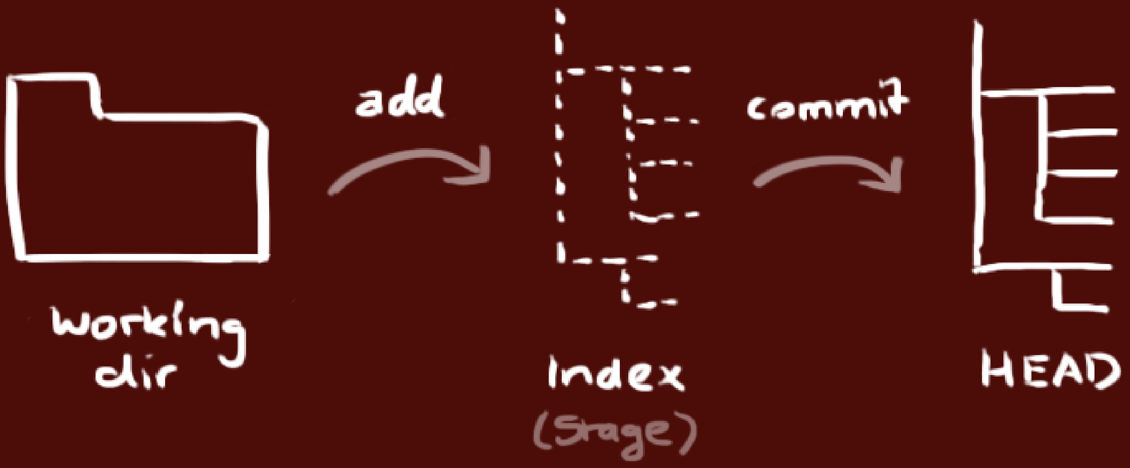
```
git clone /path/to/repository
```

when using a remote server, your command will be

```
git clone username@host:/path/to/repository
```

workflow

your local repository consists of three "trees" maintained by git. the first one is your `Working Directory` which holds the actual files. the second one is the `Index` which acts as a staging area and finally the `HEAD` which points to the last commit you've made.



add & commit

You can propose changes (add it to the **Index**) using

```
git add <filename>
```

```
git add *
```

This is the first step in the basic git workflow. To actually commit these changes use

```
git commit -m "Commit message"
```

Now the file is committed to the **HEAD**, but not in your remote repository yet.

pushing changes

Your changes are now in the **HEAD** of your local working copy. To send those changes to your remote repository, execute

```
git push origin master
```

Change *master* to whatever branch you want to push your changes to.

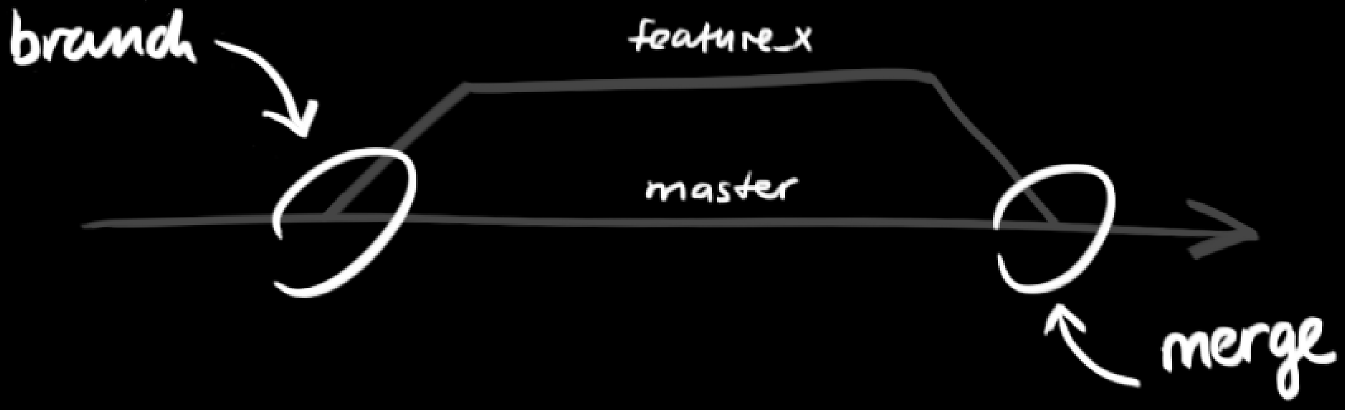
If you have not cloned an existing repository and want to connect your repository to a remote server, you need to add it with

```
git remote add origin <server>
```

Now you are able to push your changes to the selected remote server

branching

Branches are used to develop features isolated from each other. The *master* branch is the "default" branch when you create a repository. Use other branches for development and merge them back to the master branch upon completion.



create a new branch named "feature_x" and switch to it using

```
git checkout -b feature_x
```

switch back to master

```
git checkout master
```

and delete the branch again

```
git branch -d feature_x
```

a branch is *not available to others* unless you push the branch to your remote repository

```
git push origin <branch>
```

update & merge

to update your local repository to the newest commit, execute

```
git pull
```

in your working directory to *fetch* and *merge* remote changes.

to merge another branch into your active branch (e.g. master), use

```
git merge <branch>
```

in both cases git tries to auto-merge changes. Unfortunately, this is not always possible and results in *conflicts*. You are responsible to merge those *conflicts* manually by editing the files shown by git. After changing, you need to mark them as merged with

```
git add <filename>
```

before merging changes, you can also preview them by using

```
git diff <source_branch> <target_branch>
```

tagging

it's recommended to create tags for software releases. this is a known concept, which also exists in SVN. You can create a new tag named *1.0.0* by executing

```
git tag 1.0.0 1b2e1d63ff
```

the *1b2e1d63ff* stands for the first 10 characters of the commit id you want to reference with your tag. You can get the commit id by looking at the...

log

in its simplest form, you can study repository history using.. `git log`
You can add a lot of parameters to make the log look like what you want.
To see only the commits of a certain author:

```
git log --author=bob
```

To see a very compressed log where each commit is one line:

```
git log --pretty=oneline
```

Or maybe you want to see an ASCII art tree of all the branches, decorated with the names of tags and branches:

```
git log --graph --oneline --decorate --all
```

See only which files have changed:

```
git log --name-status
```

These are just a few of the possible parameters you can use. For more, see `git log --help`

replace local changes

In case you did something wrong, which for sure never happens ;), you can replace local changes using the command

```
git checkout -- <filename>
```

this replaces the changes in your working tree with the last content in HEAD. Changes already added to the index, as well as new files, will be kept.

If you instead want to drop all your local changes and commits, fetch the latest history from the server and point your local master branch at it like this

```
git fetch origin
```

```
git reset --hard origin/master
```

useful hints

built-in git GUI

```
gitk
```

use colorful git output

```
git config color.ui true
```

show log on just one line per commit

```
git config format.pretty oneline
```

use interactive adding

```
git add -i
```

links & resources

graphical clients

GitX (L) (OSX, open source)
Tower (OSX)
Source Tree (OSX & Windows, free)
GitHub for Mac (OSX, free)
GitBox (OSX, App Store)

guides

Git Community Book
Pro Git
Think like a git
GitHub Help
A Visual Git Guide

get help

Git User Mailing List
#git on irc.freenode.net

comments

533 Comments   git - the simple guide   1 Login ▾

♥ Recommend 150   Share   Sort by Newest ▾

Join the discussion…

Mikhail Zoupas • 2 days ago
What a life saver !! Thanks thanks thanks
1 ⌃ | ⌄ • Reply • Share ›

agustin cahyani • 2 days ago
omg... this tutorial saved my life. Thank you so much
1 ⌃ | ⌄ • Reply • Share ›

Kalhan Liyanage • 3 days ago
,simple , understandable and straight to the point. Bunch of thanks !!
1 ⌃ | ⌄ • Reply • Share ›

Biny Yawitz • 3 days ago

one of the useful and direct to target guide!

⌃ | ⌄ • Reply • Share ›

Vitor R Zanatta • 3 days ago

Excellent guide!

⌃ | ⌄ • Reply • Share ›

KennethB • 4 days ago

Well done.

"brevity is the soul of wit"
-- Shakespeare, Hamlet Act 2, scene 2

⌃ | ⌄ • Reply • Share ›

mutunake • 7 days ago

perfect tutorial... why most need to complicate things so much when they are so simple?!?!

1 ⌃ | ⌄ • Reply • Share ›

Chow Lung • 8 days ago

Great guide, thanks.

⌃ | ⌄ • Reply • Share ›

Toni • 15 days ago

Great! Nice job

⌃ | ⌄ • Reply • Share ›

Robert Mckeown • 15 days ago

Nice...elegant and simple..thx

1 ⌃ | ⌄ • Reply • Share ›

Fabricio Mariani • 16 days ago

Thanks! It helped me

1 ⌃ | ⌄ • Reply • Share ›

Kirshan • 17 days ago

thanks, this is good kick start :)

⌃ | ⌄ • Reply • Share ›

Tung • 18 days ago

Thank you so much for this simple guide. That's perfect for me.

⌃ | ⌄ • Reply • Share ›

Vic • 19 days ago

Awesome starting guide, no deep stuff

⌃ | ⌄ • Reply • Share ›

Shreekara Tantry • 20 days ago

you are awesome baby!

⌃ | ⌄ • Reply • Share ›

Mario Botero • 21 days ago

Thanks a good resume of the principal commands

⌃ | ⌄ • Reply • Share ›

Mohamed Zaatar • 22 days ago

nice guide but with workflows diagrams will be perfect.

⌃ | ⌄ • Reply • Share ›

Sebastian • 24 days ago

Awesome!

^ | ˅ • Reply • Share ›

Md Imran Mondal • 25 days ago
love this guide, well explained
^ | ˅ • Reply • Share ›

Mbah-Mbole Sama • 25 days ago
great tutorial, short concise and elaborate. Thank you!
^ | ˅ • Reply • Share ›

Yarina • a month ago
Love this!
^ | ˅ • Reply • Share ›

schlumpfmarkus • a month ago
Super simple and to the point! Love the colors. Thank a lot!!
^ | ˅ • Reply • Share ›

Ralf Tenbrink • a month ago
I cannot get past step 2 (clone). I get permission denied (publickey).
Was hoping this would help me understand github but I cannot get my head around it. I better stick with SVN
^ | ˅ • Reply • Share ›

Nils Mundhenke • a month ago
This is the best thing I saw this year!
Thank you so much for this :)
1 ^ | ˅ • Reply • Share ›

devang nathwani • a month ago
thank you. thank you.
1 ^ | ˅ • Reply • Share ›

PHORN Ya • a month ago
The best tutorial for beginner
1 ^ | ˅ • Reply • Share ›

Sree Kumar • a month ago
Really awesome !!!
^ | ˅ • Reply • Share ›

devesh • a month ago
how to merge change made in one branch to another
^ | ˅ • Reply • Share ›

sirslimedia ➦ devesh • a month ago
git merge <branch2> e.g git merge updates and you can check the diff with git diff branch1 branch2
^ | ˅ • Reply • Share ›

Rajat Pawar • a month ago
Awesome tutorial !
^ | ˅ • Reply • Share ›

Barnaby Jones • a month ago
A great but still quite complex tutorial. How do you know what server to connect to? Doesn't this happen automatically if you use Github? How does git automatically know your username and email? Where are these stored?
1 ^ | ˅ • Reply • Share ›

Chow Lung ➦ Barnaby Jones • 8 days ago

Hi Barnaby,

When you call:

git clone username@host:path/to/your/repository

You provide both a username and host.
It will then prompt you for a password.

So it's not "automatic" you provide the necessary information.
I hope that's resolved some of your questions.

note: You can likely store/save this information for repeat use, however I couldn't tell you how. This is my first day using git.

2 ^ | ˅ • Reply • Share ›

Rajat Pawar ➦ Barnaby Jones • a month ago

This tutorial is intended for software engineers.

^ | ˅ • Reply • Share ›

Chytap Richfriend • a month ago

Thank you very much

^ | ˅ • Reply • Share ›

kuldipem • a month ago

nice quick tutorials....

^ | ˅ • Reply • Share ›

carlos enrique olivares rodrig • a month ago

Excelent very helpfull. I just migrate from subversion to git and this guide is exactly what i need. Thanks Roger & Nina

^ | ˅ • Reply • Share ›

Kaia Konsap • a month ago

I LOVE this page, thank you!

^ | ˅ • Reply • Share ›

Eduardo Adrian Perez • a month ago

Excelent !

^ | ˅ • Reply • Share ›

Dennis Okparaocha • a month ago

Just what I needed. thanks :)

^ | ˅ • Reply • Share ›

Nerunjakumar Soubaya • a month ago

Very useful and handy. Thanks. I think, command to delete a file and also how to get it back could you added here.

^ | ˅ • Reply • Share ›

David Méndez Acuña • a month ago

Just great! Thanks a lot!

^ | ˅ • Reply • Share ›

Ryan Knutson • a month ago

This is wonderful! I now understand git! Thank you!

^ | ˅ • Reply • Share ›

Febriyant Abidin • a month ago

nice web guide dude

^ | ˅ • Reply • Share ›

karthick • a month ago

Awsome post, I had one issue with pull files from server, I am created a new branch in gitlab and pushed some files to my new repo , and then i get logged in with ssh and get some other files from other server through wget method , the files are get stored in server , and files are showing in browser , but when i pull the files it not getting into my localrepo , every i remove and again clone the repo but still i am not able to get the files which is get with wget method . Can someone help me on this.

^ | ⌄ • Reply • Share ›

Arunvel Sriram • a month ago

Simply explained. Very useful. Thank you.

^ | ⌄ • Reply • Share ›

Sriram B • a month ago

Really awesome !!! Simple and Useful :)

^ | ⌄ • Reply • Share ›

WalkerHaleIV • a month ago

Note that the download link for Mac OS X is out of date, currently gives an obsolete version, and will break when Google Code shuts down in 2016. Instead go here: http://git-scm.com/download/ma...

^ | ⌄ • Reply • Share ›

ameya • a month ago

Would have been better if you would have should every steps on gitbash/github with example.

^ | ⌄ • Reply • Share ›

Anubha Kumari • a month ago

Excellent concept! I've featured it on Hackr.io's Git section as well - http://hackr.io/tutorials/git

^ | ⌄ • Reply • Share ›

girish • 2 months ago

great work guys..

^ | ⌄ • Reply • Share ›

Load more comments

ALSO ON **GIT - THE SIMPLE GUIDE**

WHAT'S THIS?

git - basit rehber - atla deve değil!
6 comments • 2 years ago

git - prosty przewodnik - nic skomplikowanego!
1 comment • 3 months ago

git - panduan ringkas - gak pake ribet!
1 comment • 3 months ago

git - een simpele uitleg - no deep shit!
8 comments • 2 years ago